SpyWolf

Project Audit

Project:

Solmit

May 13, 2022

Solmit

Overview

This audit has been prepared for **Solmit** to review the main aspects of the project to help investors make an informative decision in the research process.

You will find a a summarized review of the following main key points:

- Contract's source code
- Project and team
- Website
- Social media & online presence

NOTE: We ONLY consider a project safe if they receive our "Certificate of Trust" NFT. This report only points out any potential red flags found in our analysis. Always do your own research before investing in a project.

Smart Contract Review

The contract review process pays special attention to the following:

- Testing the smart contracts against both common and uncommon vulnerabilities
- Assessing the codebase to ensure compliance with current best practices and industry standards.
- Ensuring contract logic meets the specifications and intentions of the client.
- Cross referencing contract structure and implementation against similar smart contracts produced by industry leaders.
- Thorough line-by-line manual review of the entire codebase by industry experts.

"The results of this audit are purely based on the team's evaluation and does not guarantee nor reflect the projects outcome and goal"
- SpyWolf Team

Smart Contract Summary

Contract Name	Solmit
Ticker	SOLT
Contract	0xE428B26Fb77c6D7aC41c8bE2D36F74962Ce8bc3e
Network	Binance smart chain
Language	Solidity
Tax	**None**
Total Supply	2,000,000,000
Status	Not launched

Current stats

Burn	No tokens burnt
Liquidity	Liquidity not added yet
MaxTxAmount	No limit
Liquidity presale percent allocation	15%

Issues Checking Status	
Design Logic	Passed ✓
Compiler warnings.	Passed ✓
Private user data leaks	Passed ✓
Timestamp dependence	Passed ✓
Integer Overflow and Underflow	Passed ✓
Race conditions and Reentrancy. Cross-function race conditions	Passed ✓
Possible delays in data delivery	Passed ✓
Oracle calls	Passed ✓
Front running	Passed ✓
DoS with Revert	Passed ✓
DoS with block gas limit	Passed ✓
Methods execution permissions	Passed ✓
Economy model	Passed ✓
The impact of the exchange rate on the logic	Passed ✓
Malicious Event log	Passed ✓
Scoping and Declarations	Passed ✓
Uninitialized storage pointers	Passed ✓
Arithmetic accuracy	Passed ✓
Cross-function race conditions	Passed ✓
Safe Zeppelin module	Passed ✓
Fallback function security	Passed ✓

Featured Wallets

Owner address	0x82e69F1c92ef28ff77241060580202B78AD935b5
*BUSD receive	0x82e69F1c92ef28ff77241060580202B78AD935b5
LP address	Liquidity not added yet

*Owner can change this address in future

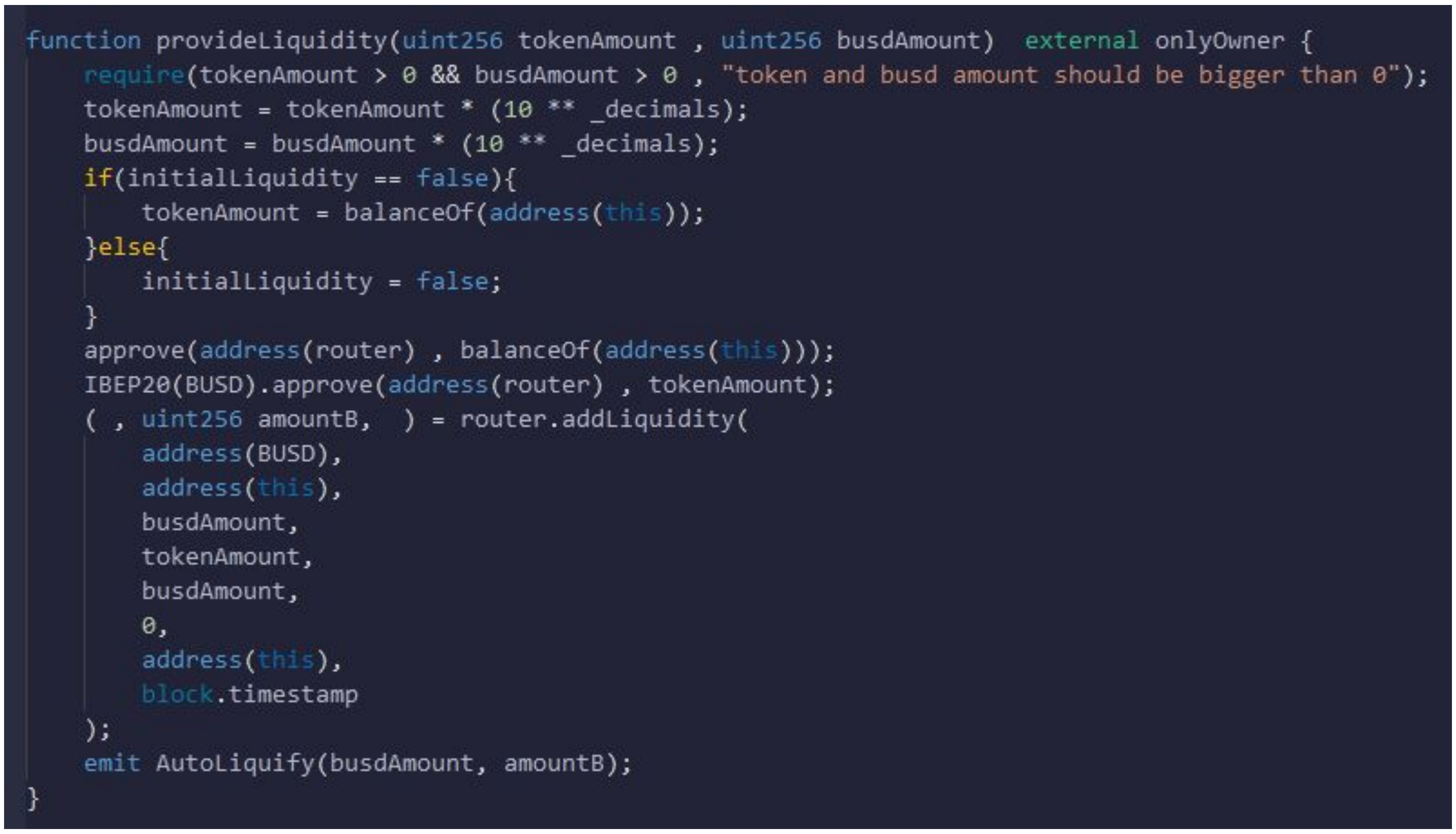
Top 3 Unlocked Wallets

Wallet 1 (100%)	Same as owner

Tokens are not distributed yet

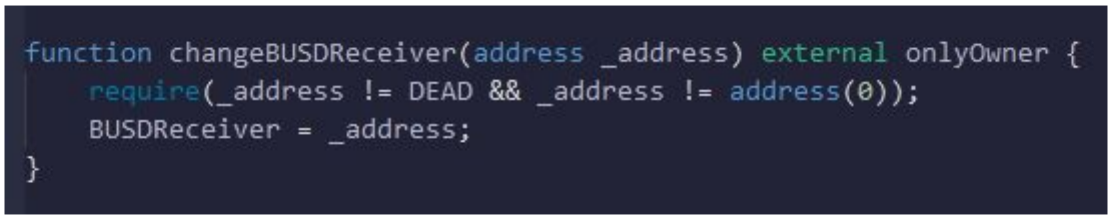
Security Threats

⚠️ Liquidity added with the provideLiquidity() function will be unlocked and can be withdrawn at any time.



```
function provideLiquidity(uint256 tokenAmount , uint256 busdAmount)  external onlyOwner {
    require(tokenAmount > 0 && busdAmount > 0 , "token and busd amount should be bigger than 0");
    tokenAmount = tokenAmount * (10 ** _decimals);
    busdAmount = busdAmount * (10 ** _decimals);
    if(initialLiquidity == false){
        tokenAmount = balanceOf(address(this));
    }else{
        initialLiquidity = false;
    }
    approve(address(router) , balanceOf(address(this)));
    IBEP20(BUSD).approve(address(router) , tokenAmount);
    ( , uint256 amountB,  ) = router.addLiquidity(
        address(BUSD),
        address(this),
        busdAmount,
        tokenAmount,
        busdAmount,
        0,
        address(this),
        block.timestamp
    );
    emit AutoLiquify(busdAmount, amountB);
}
```

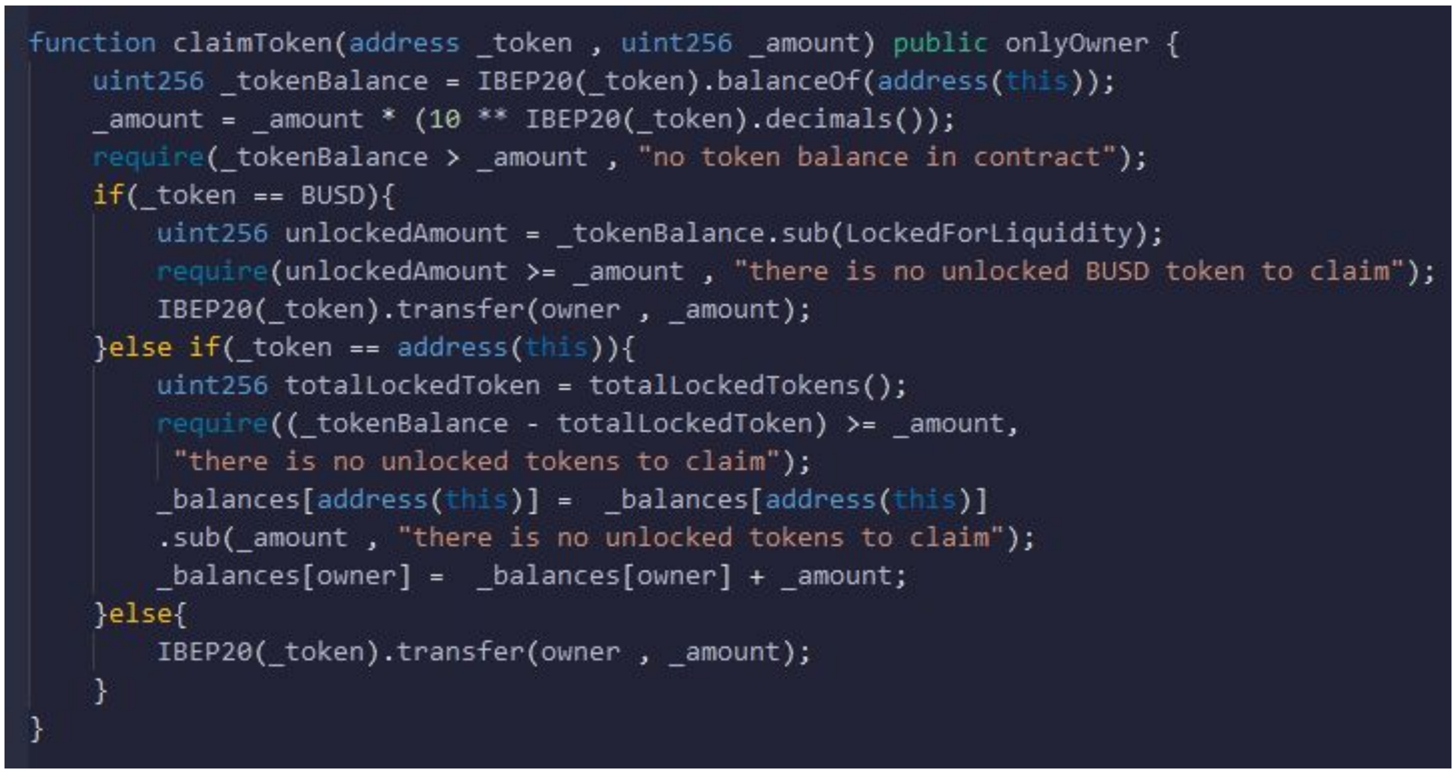
When presalers buy tokens from the contract, BUSD goes into the BUSDReceiver address. Owner can change BUSDReceiver to any wallet.



```
function changeBUSDReceiver(address _address) external onlyOwner {
    require(_address != DEAD && _address != address(0));
    BUSDReceiver = _address;
}
```

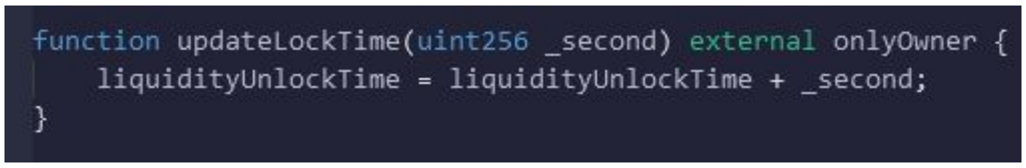
Security Threats

Owner can withdraw any tokens from the contract.



```
function claimToken(address _token , uint256 _amount) public onlyOwner {
    uint256 _tokenBalance = IBEP20(_token).balanceOf(address(this));
    _amount = _amount * (10 ** IBEP20(_token).decimals());
    require(_tokenBalance > _amount , "no token balance in contract");
    if(_token == BUSD){
        uint256 unlockedAmount = _tokenBalance.sub(LockedForLiquidity);
        require(unlockedAmount >= _amount , "there is no unlocked BUSD token to claim");
        IBEP20(_token).transfer(owner , _amount);
    }else if(_token == address(this)){
        uint256 totalLockedToken = totalLockedTokens();
        require((_tokenBalance - totalLockedToken) >= _amount,
         "there is no unlocked tokens to claim");
        _balances[address(this)] =  _balances[address(this)]
        .sub(_amount , "there is no unlocked tokens to claim");
        _balances[owner] =  _balances[owner] + _amount;
    }else{
        IBEP20(_token).transfer(owner , _amount);
    }
}
```

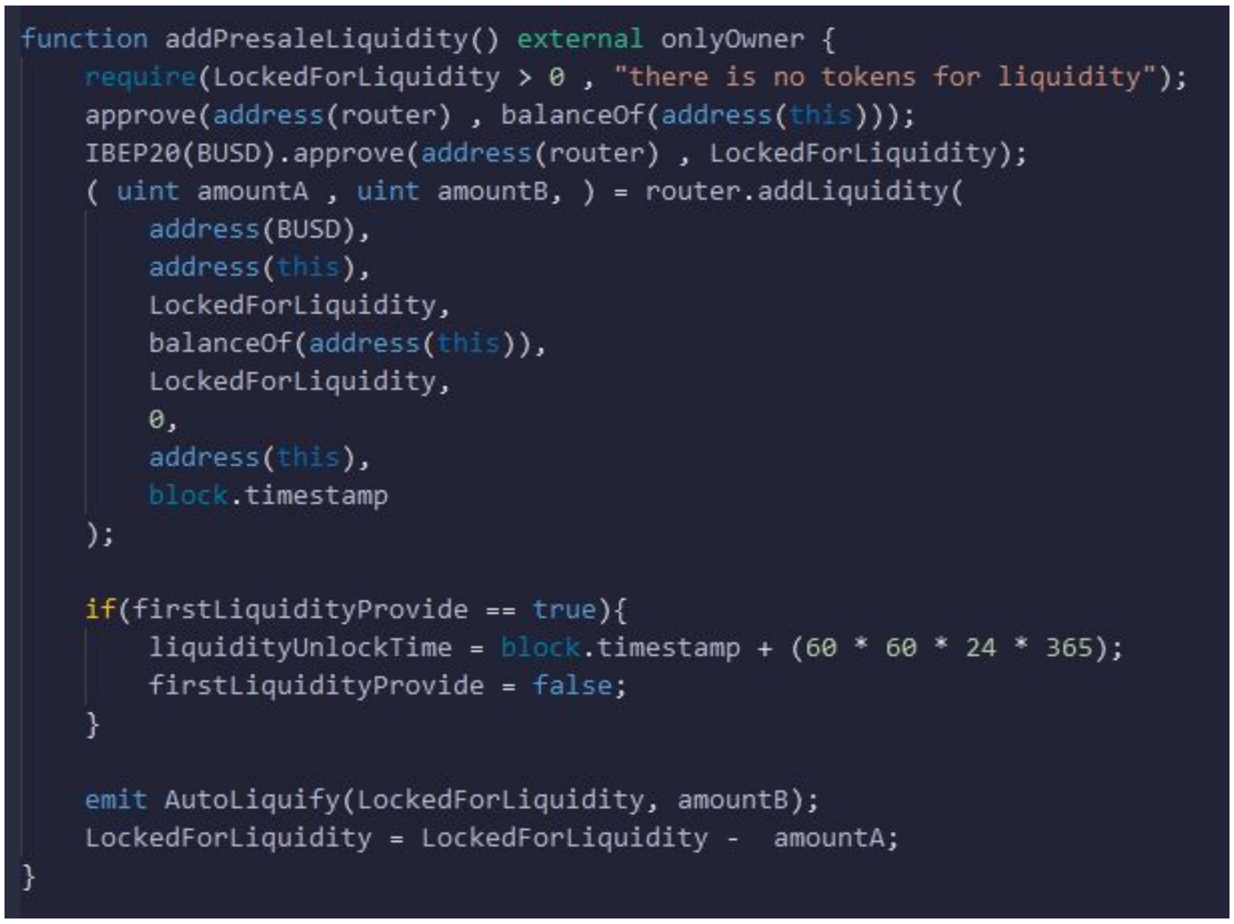
Owner can extend the liquidity lock time.



```
function updateLockTime(uint256 _second) external onlyOwner {
    liquidityUnlockTime = liquidityUnlockTime + _second;
}
```

Security Threats

Liquidity added with the addPresaleLiquidity() function will be locked for a year.

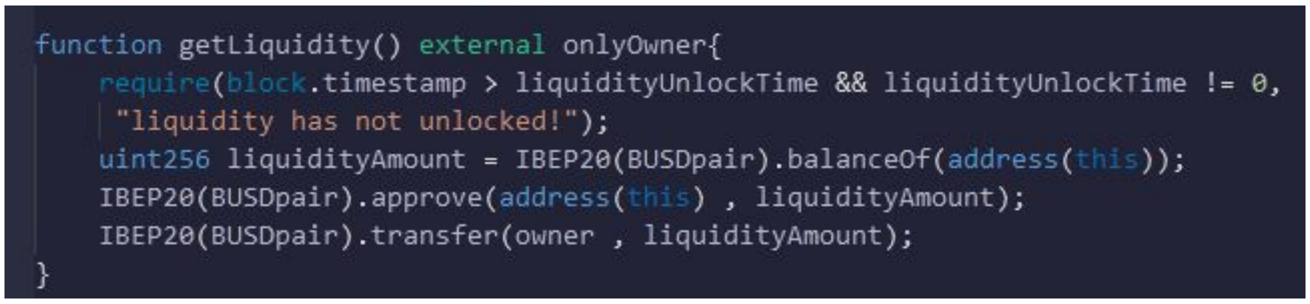


```
function addPresaleLiquidity() external onlyOwner {
    require(LockedForLiquidity > 0 , "there is no tokens for liquidity");
    approve(address(router) , balanceOf(address(this)));
    IBEP20(BUSD).approve(address(router) , LockedForLiquidity);
    ( uint amountA , uint amountB, ) = router.addLiquidity(
        address(BUSD),
        address(this),
        LockedForLiquidity,
        balanceOf(address(this)),
        LockedForLiquidity,
        0,
        address(this),
        block.timestamp
    );

    if(firstLiquidityProvide == true){
        liquidityUnlockTime = block.timestamp + (60 * 60 * 24 * 365);
        firstLiquidityProvide = false;
    }

    emit AutoLiquify(LockedForLiquidity, amountB);
    LockedForLiquidity = LockedForLiquidity -  amountA;
}
```

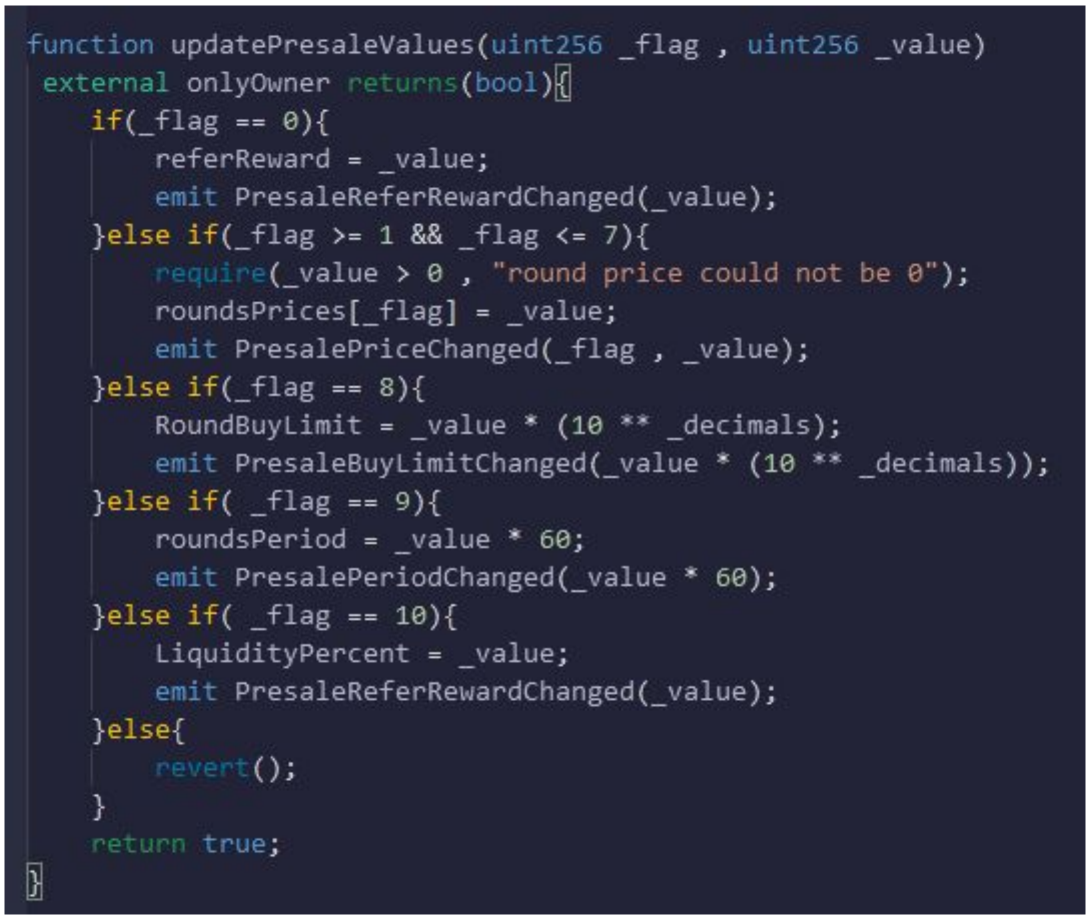
Owner can’t withdraw the liquidity, untill the liquidity unlock time is expired.



```
function getLiquidity() external onlyOwner{
    require(block.timestamp > liquidityUnlockTime && liquidityUnlockTime != 0,
     "liquidity has not unlocked!");
    uint256 liquidityAmount = IBEP20(BUSDpair).balanceOf(address(this));
    IBEP20(BUSDpair).approve(address(this) , liquidityAmount);
    IBEP20(BUSDpair).transfer(owner , liquidityAmount);
}
```

Security Threats

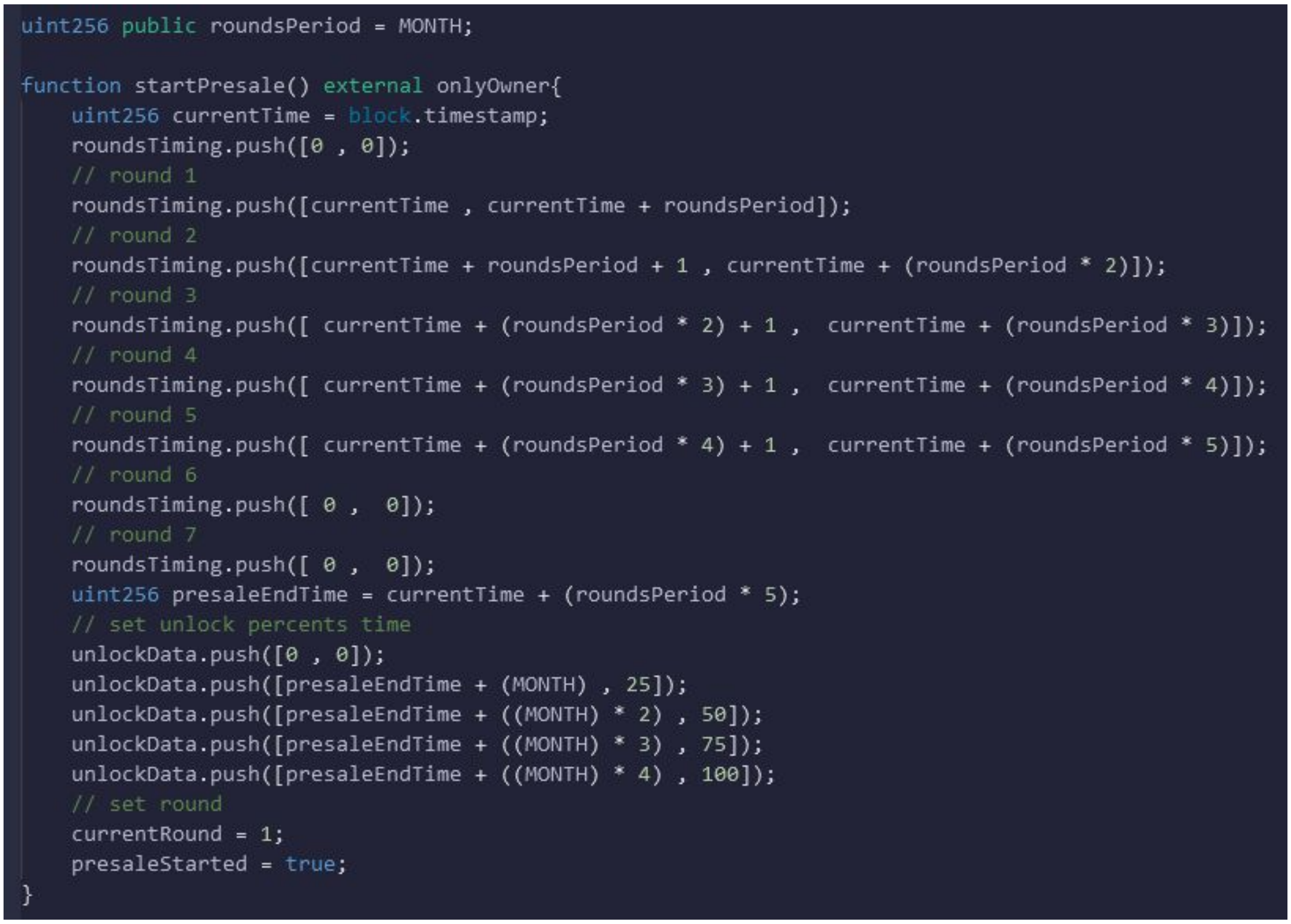
Owner can change the following presale settings:
Referer reward percent.
Current round tokens presale price.
Each round duration period.
Tokens allocated for liquidity percent.
Tokens available for presale for each round.



```
function updatePresaleValues(uint256 _flag , uint256 _value)
 external onlyOwner returns(bool){
    if(_flag == 0){
        referReward = _value;
        emit PresaleReferRewardChanged(_value);
    }else if(_flag >= 1 && _flag <= 7){
        require(_value > 0 , "round price could not be 0");
        roundsPrices[_flag] = _value;
        emit PresalePriceChanged(_flag , _value);
    }else if(_flag == 8){
        RoundBuyLimit = _value * (10 ** _decimals);
        emit PresaleBuyLimitChanged(_value * (10 ** _decimals));
    }else if( _flag == 9){
        roundsPeriod = _value * 60;
        emit PresalePeriodChanged(_value * 60);
    }else if( _flag == 10){
        LiquidityPercent = _value;
        emit PresaleReferRewardChanged(_value);
    }else{
        revert();
    }
    return true;
}
```

Security Threats

⚠️ Owner can activate startPresale() function more than once, causing prolonging of user's vested time schedule.
Presale tokens vesting unlock periods are as follows:
25% of bought tokens in 6 months.
50% of bought tokens in 7 months.
75% of bought tokens in 8 months.
100% of bought tokens in 9 months.



```
uint256 public roundsPeriod = MONTH;

function startPresale() external onlyOwner{
    uint256 currentTime = block.timestamp;
    roundsTiming.push([0 , 0]);
    // round 1
    roundsTiming.push([currentTime , currentTime + roundsPeriod]);
    // round 2
    roundsTiming.push([currentTime + roundsPeriod + 1 , currentTime + (roundsPeriod * 2)]);
    // round 3
    roundsTiming.push([ currentTime + (roundsPeriod * 2) + 1 ,  currentTime + (roundsPeriod * 3)]);
    // round 4
    roundsTiming.push([ currentTime + (roundsPeriod * 3) + 1 ,  currentTime + (roundsPeriod * 4)]);
    // round 5
    roundsTiming.push([ currentTime + (roundsPeriod * 4) + 1 ,  currentTime + (roundsPeriod * 5)]);
    // round 6
    roundsTiming.push([ 0 ,  0]);
    // round 7
    roundsTiming.push([ 0 ,  0]);
    uint256 presaleEndTime = currentTime + (roundsPeriod * 5);
    // set unlock percents time
    unlockData.push([0 , 0]);
    unlockData.push([presaleEndTime + (MONTH) , 25]);
    unlockData.push([presaleEndTime + ((MONTH) * 2) , 50]);
    unlockData.push([presaleEndTime + ((MONTH) * 3) , 75]);
    unlockData.push([presaleEndTime + ((MONTH) * 4) , 100]);
    // set round
    currentRound = 1;
    presaleStarted = true;
}
```

Smart Contract Summary

Contract Name	Solmit
Ticker	SOLT
Contract	0xE428B26Fb77c6D7aC41c8bE2D36F74962Ce8bc3e
Network	Polygon
Language	Solidity
Tax	**None**
Total Supply	2,000,000,000
Status	Not launched

Current stats

Burn	No tokens burnt
LP Address	0xF90d283A011e7e805BCAe09AaF7D7Ef0Dcba6402
Liquidity	261 WMATIC
MaxTxAmount	No limit

Featured Wallets

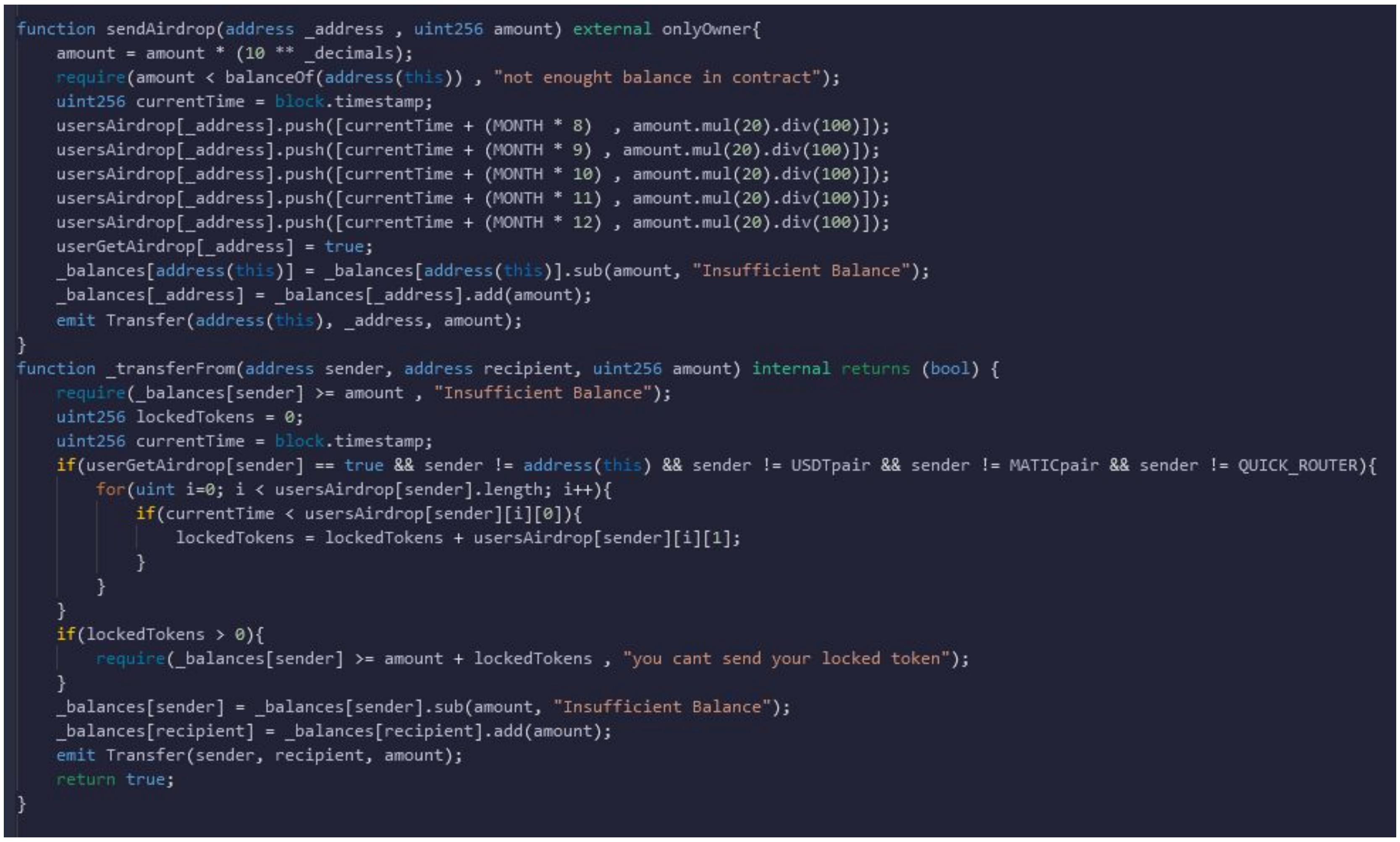
Owner address	0x82e69F1c92ef28ff772410605802O2B78AD935b5
LP address	0xF90d283A011e7e805BCAe09AaF7D7Ef0Dcba6402

Top 3 Unlocked Wallets

Wallet 1 (94.99%)	Same as owner
Wallet 2 (4.97%)	0xE428B26Fb77c6D7aC41c8bE2D36F74962Ce8bc3e **Solmit contract**
Wallet 3 (0.0138%)	0x45b9c7bc9b826635580502a1d5399217050983c7

Security Threats

The airdropped tokens can be claimed in the following vesting periods:
20% of airdropped tokens after 8 months
20% of airdropped tokens after 9 months
20% of airdropped tokens after 10 months
20% of airdropped tokens after 11 months
20% of airdropped tokens after 12 months



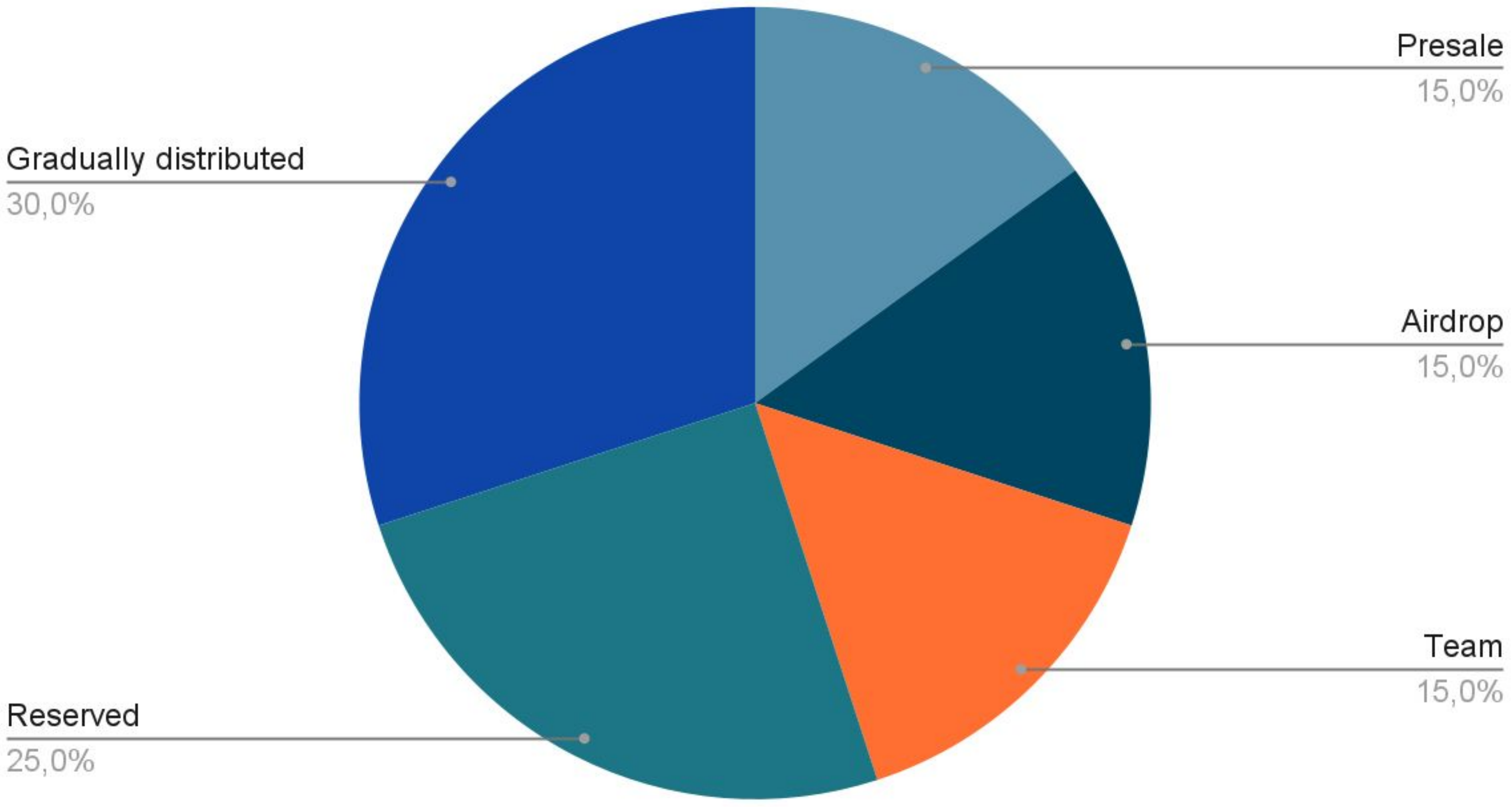
```
function sendAirdrop(address _address , uint256 amount) external onlyOwner{
    amount = amount * (10 ** _decimals);
    require(amount < balanceOf(address(this)) , "not enought balance in contract");
    uint256 currentTime = block.timestamp;
    usersAirdrop[_address].push([currentTime + (MONTH * 8)  , amount.mul(20).div(100)]);
    usersAirdrop[_address].push([currentTime + (MONTH * 9) , amount.mul(20).div(100)]);
    usersAirdrop[_address].push([currentTime + (MONTH * 10) , amount.mul(20).div(100)]);
    usersAirdrop[_address].push([currentTime + (MONTH * 11) , amount.mul(20).div(100)]);
    usersAirdrop[_address].push([currentTime + (MONTH * 12) , amount.mul(20).div(100)]);
    userGetAirdrop[_address] = true;
    _balances[address(this)] = _balances[address(this)].sub(amount, "Insufficient Balance");
    _balances[_address] = _balances[_address].add(amount);
    emit Transfer(address(this), _address, amount);
}
function _transferFrom(address sender, address recipient, uint256 amount) internal returns (bool) {
    require(_balances[sender] >= amount , "Insufficient Balance");
    uint256 lockedTokens = 0;
    uint256 currentTime = block.timestamp;
    if(userGetAirdrop[sender] == true && sender != address(this) && sender != USDTpair && sender != MATICpair && sender != QUICK_ROUTER){
        for(uint i=0; i < usersAirdrop[sender].length; i++){
            if(currentTime < usersAirdrop[sender][i][0]){
                lockedTokens = lockedTokens + usersAirdrop[sender][i][1];
            }
        }
    }
    if(lockedTokens > 0){
        require(_balances[sender] >= amount + lockedTokens , "you cant send your locked token");
    }
    _balances[sender] = _balances[sender].sub(amount, "Insufficient Balance");
    _balances[recipient] = _balances[recipient].add(amount);
    emit Transfer(sender, recipient, amount);
    return true;
}
```

Tokenomics

According to their whitepaper:

- 15% - Presale at 3 stages
- 15% - Airdrop
- 15% - Team
- 25% - Reserved to build blockchain
- 30% - Gradually distributed to SOLMIT community

Tokens distribution



Solmit
Project & Team Review

According to their whitepaper:
SOLMIT's field of activity will be mainly within the insurance and financial services industries. SOLMIT platform will operate in three areas of insurance:

- Investment Insurance
- Life Insurance
- Health Insurance

The project's future development will be as follows:

- NFT marketplace
- Solmit SWAP

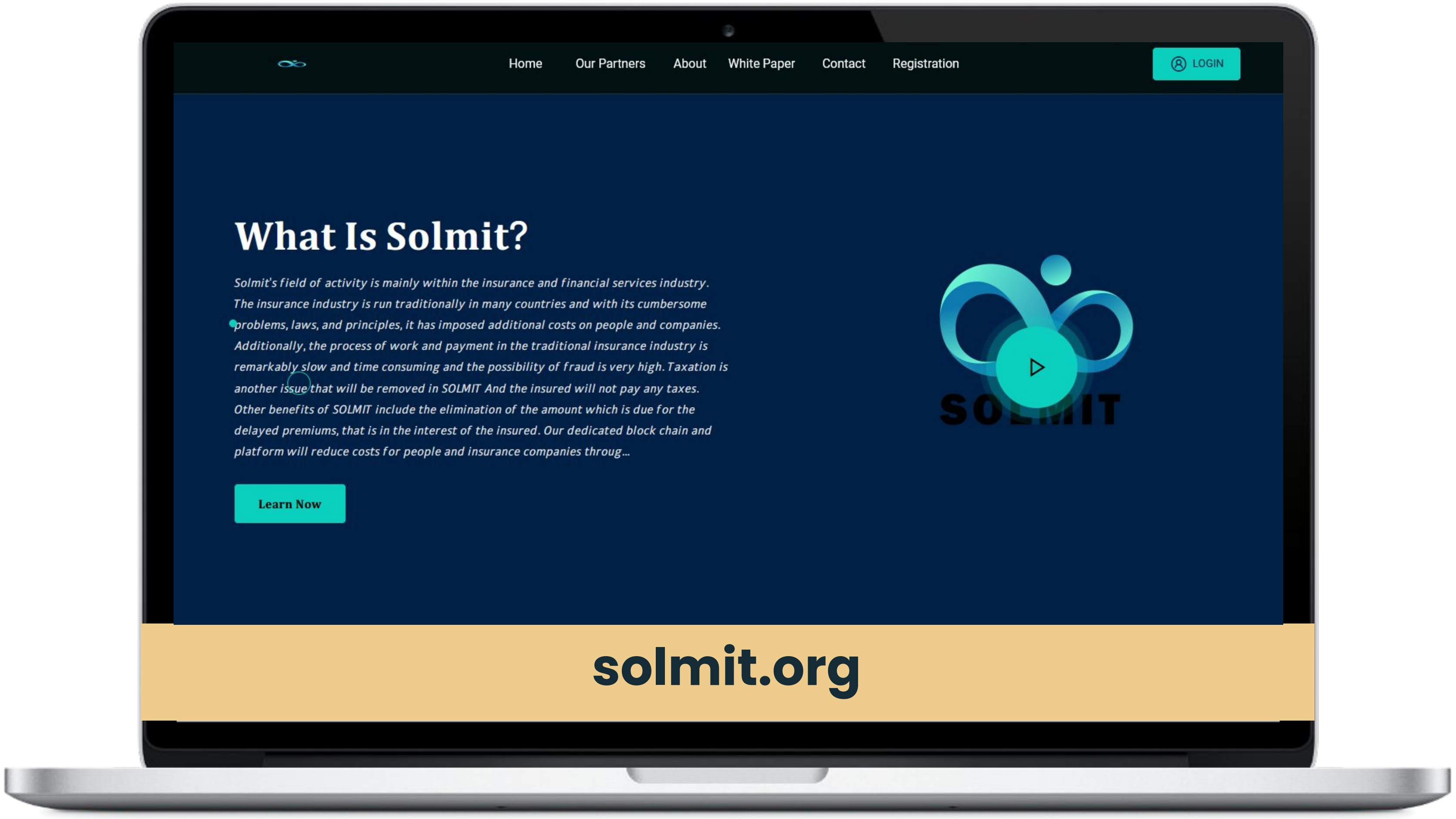
Team:

⚠️ **Team has not been KYC'd** ⚠️

Website Analysis

URL: https://solmit.org/

- **Design:** Nice single page design, appropriate color scheme.
- **Content:** Confusing, general information without specifics. ⚠️
- **Whitepaper:** Confusing, general information without specifics, no clear explanation how insurance mechanisms will work. ⚠️
- **Roadmap:** Goals set at 6 phases without any time frames.
- **Mobile-friendly?** Yes
- **Technical:** SSL certificate present. General SEO check passed.



What Is Solmit?

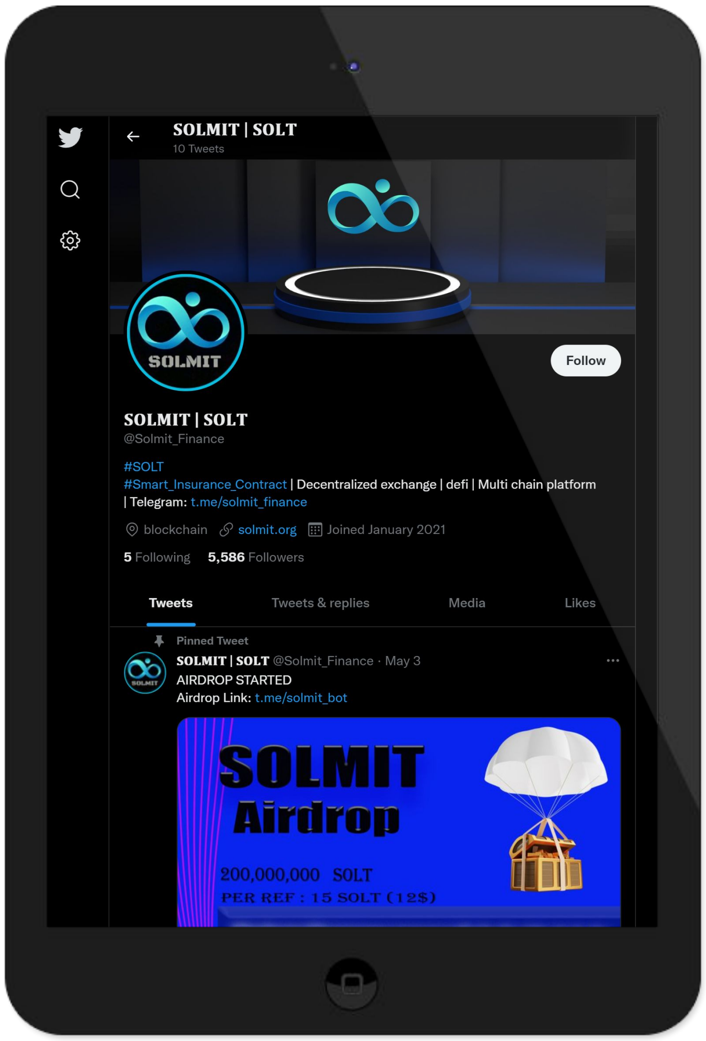
Solmit's field of activity is mainly within the insurance and financial services industry. The insurance industry is run traditionally in many countries and with its cumbersome problems, laws, and principles, it has imposed additional costs on people and companies. Additionally, the process of work and payment in the traditional insurance industry is remarkably slow and time consuming and the possibility of fraud is very high. Taxation is another issue that will be removed in SOLMIT And the insured will not pay any taxes. Other benefits of SOLMIT include the elimination of the amount which is due for the delayed premiums, that is in the interest of the insured. Our dedicated block chain and platform will reduce costs for people and insurance companies throug...

Social Media & Online Presence

Telegram

https://t.me/solmit_finance

- 11 528 members
- Announcement channel
- No chat ⚠️



SOLMIT | SOLT
10 Tweets

SOLMIT | SOLT
@Solmit_Finance

#SOLT
#Smart_Insurance_Contract | Decentralized exchange | defi | Multi chain platform | Telegram: t.me/solmit_finance

blockchain solmit.org Joined January 2021

5 Following 5,586 Followers

SOLMIT | SOLT @Solmit_Finance · May 3
AIRDROP STARTED
Airdrop Link: t.me/solmit_bot

200,000,000 SOLT
PER REF : 15 SOLT (12$)

Twitter

https://twitter.com/solmit_finance

- 5 568 Followers
- Only 6 posts (3 of them made in 1 day) ⚠️

About SpyWolf

SpyWolf is a team of crypto security experts that have been performing full audits for projects for the past months in order to ensure safety on the crypto space. Our goal is to help eliminate monetary fraud through our auditing services and utility token, $SPY.

- Website: SpyWolf.co
- Portal: SpyWolf.network
- Telegram: @SpyWolfNework
- Twitter: Twitter.com/SpyWolfNetwork

(Sample Certificate NFT for those who pass audit)

If you are interested in finding out more about our audits and Certificate of Trust NFTs, reach out to contact@spywolf.co.

Disclaimer

This report shows findings based on our limited project analysis, following good industry practice from the date of this report, in relation to cybersecurity vulnerabilities and issues in the framework and algorithms based on smart contracts, overall social media and website presence and team transparency details of which are set out in this report. In order to get a full view of our analysis, it is crucial for you to read the full report.
While we have done our best in conducting our analysis and producing this report, it is important to note that you should not rely on this report and cannot claim against us on the basis of what it says or doesn't say, or how we produced it, and it is important for you to conduct your own independent investigations before making any decisions. We go into more detail on this in the disclaimer below – please make sure to read it in full.

DISCLAIMER:
By reading this report or any part of it, you agree to the terms of this disclaimer. If you do not agree to the terms, then please immediately cease reading this report, and delete and destroy any and all copies of this report downloaded and/or printed by you. This report is provided for information purposes only and on a non-reliance basis, and does not constitute investment advice.
No one shall have any right to rely on the report or its contents, and SpyWolf and its affiliates (including holding companies, shareholders, subsidiaries, employees, directors, officers and other representatives) (SpyWolf) owe no duty of care towards you or any other person, nor does SpyWolf make any warranty or representation to any person on the accuracy or completeness of the report.
The report is provided "as is", without any conditions, warranties or other terms of any kind except as set out in this disclaimer, and SpyWolf hereby excludes all representations, warranties, conditions and other terms (including, without limitation, the warranties implied by law of satisfactory quality, fitness for purpose and the use of reasonable care and skill) which, but for this clause, might have effect in relation to the report. Except and only to the extent that it is prohibited by law, SpyWolf hereby excludes all liability and responsibility, and neither you nor any other person shall have any claim against SpyWolf, for any amount or kind of loss or damage that may result to you or any other person (including without limitation, any direct, indirect, special, punitive, consequential or pure economic loss or damages, or any loss of income, profits, goodwill, data, contracts, use of money, or business interruption, and whether in delict, tort (including without limitation negligence), contract, breach of statutory duty, misrepresentation (whether innocent or negligent) or otherwise under any claim of any nature whatsoever in any jurisdiction) in any way arising from or connected with this report and the use, inability to use or the results of use of this report, and any reliance on this report. The analysis of the security is purely based on the smart contracts, website, social media and team.
No applications were reviewed for security. No product code has been reviewed.